Per Scintilator Module Alignment with Straight Muons

B. Viren, *BNL*

October 4, 2002

Abstract

This note documents an initial per-scintilator-module alignment using straight track muons (B-field off). The algorithm used is described and the software implementation is documented. Initial results are presented.

1 Introduction

Initially the MINOS offline software uses an idealized geometry that does not take into account any detailed real world non-uniformities. To some extent these will be taken into account using survey data[1]. To compliment as well as check this survey, straight (no B-field) cosmic ray muon tracks are used to measure any offsets in the real placement of scintilator modules from their assumed locations. The generic term "alignment" is used to label these measurements.

In this note section 2 describes the algorithm used for the alignment. Section 3 documents the software implementation. In section 4 initial results are given. Finally, some areas for future work are outlined in section 5.

2 Per Scintilator Module Alignment Algorithm

The algorithm used is an extension of one implemented previously[2] for per-plane alignment work. It works by first selecting a set of tracks passing through at least 14 detector planes. Then:

- For each track
    - For each candidate digit in the track
        - Fit a straight line using all digits in the track except the current one.

[1]Numi-NOTE-GEN-868, David Boehnlein
[2]Initial work done by Nicolai Tobien

  - ∗ Calculate the *residual* (defined below) between the fit location and the current candidate digit.
  - ∗ Collect this residual and associate it with the scintilator module containing the current candidate digit.

- Interpret the mean residual per scintilator module as a transverse offset.
- Apply these offsets to the geometry.
- Iterate.

The definition of *residual* in the above is the difference in location of the fit position at the center of the current candidate digit's plane to the center of the current candidate digit's strip. All distances are in the transverse direction for a particular view. A positive residual means that the alignment fits the location of the module to be higher than what is assumed in the initial geometry.

3 Software Implementation

The implementation exists in the MINOS offline software package "Alignment" which is included as part of current development and frozen releases.

This code consists of an **AlignmentModule** job module driver which uses a **StraightTrackAlignment** object to perform the actual calculation. The intermediate and final results are stored in histograms managed by the **AlignHists** object and further iterations occur at the end of the job.

The nature of this algorithm requires multiple passes over a set of tracks. To optimized the amount of data I/O the tracks are cached in the first pass of the data in the **StraightTrackAlignment** object.

In order to provide for multi-processing as well as check-pointing the results, the Alignment code is meant to run on a single input data file and create a single output histogram file. After multiple output histogram files are created the results are merged.

The details of running this code are as follows:

1. Generate the intermediate histograms:

   ```
   bash> loon -b -q 'align.C("data.1.root","hist.1.root")'
   ```

   This command, using the **align.C** ROOT macro file which is part of the Alignment package, will use **data.1.root** to produce intermediate histogram file **hist.1.root** file. Running on multiple input files can be facilitated with a shell script. An example of one is provided in **run-align.sh**.

   The macro file **align.C** can be modified to tell the code to use a different source for muon tracks. The default is to use the output of the CandTrackSR package.

2. Merge the intermediate histograms:

```
bash> loon -b -q 'merge.C("file.txt","hists.root")'
```

This command, using the **merge.C** file which is also part of the package, will sum the histograms in a way that preserves the correct statistics and takes into account the fact that different number of planes may exist in the intermediate histograms. The file "**file.txt**" lists each intermediate histogram file, one per line.

3. Produce default plots:

```
bash> loon -q 'plot_hists.C("hists.root")'
```

This macro gives examples of how to unpack the merged histogram file and make some simple plots of the results (see below).

4 Initial Results

The initial results used data from runs 5963, 5966, 5980, 5991, 5994, 6002, 6005, 6027, 6030, 6033, 6036, 6077, 6081, 6105, and 6109, which corresponds to 3 days of running. Figure 1 shows the detector coverage for these runs.

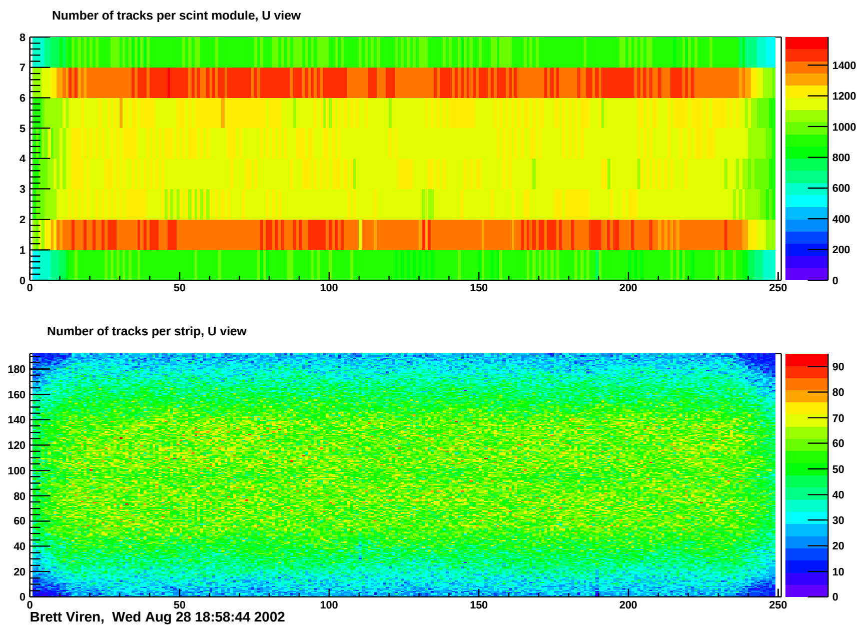

Figure 1: Number of tracks per scintilator module and per strip. Despite the labels, U and V planes are interleaved.

The detector is well covered except for the corners and edges. When plotted per scintilator module the effect of different numbers and lengths of scintilator modules is apparent.

Figures 2 – 4 show the mean and RMS of the fit residuals per scintilator module after the first, second and third iteration in scintilator module *vs.* plane.

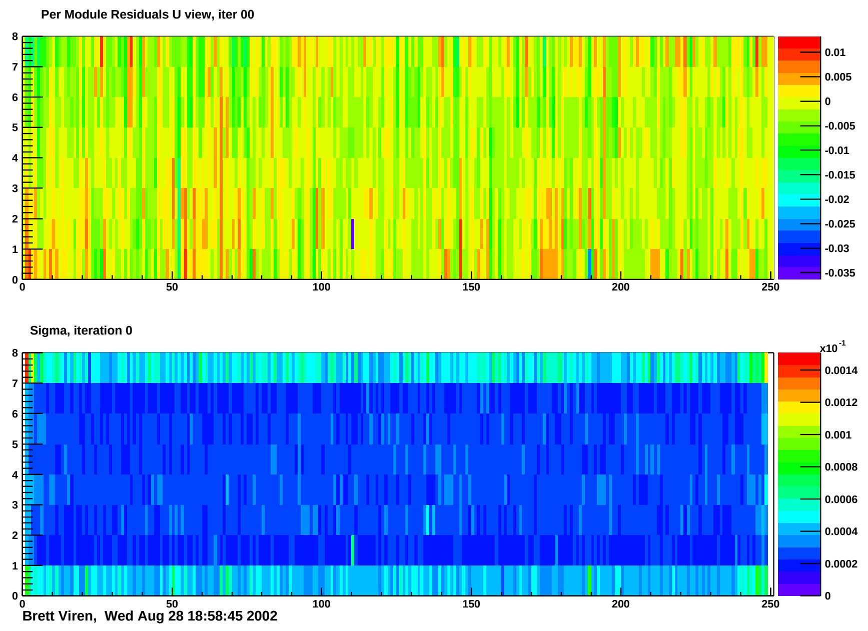

Figure 2: Per module mean and RMS fit residuals after first iteration. See text.

The color scale is given in meters. Despite the lables, the U and V views are interleaved. In the final iteration the largest negative offset is on order of 3 cm and appears in the "upper" corner of the first planes of the U and V views and the largest positive offset is on order of 1 cm in the "lower" corner. These are also areas of lower statistical coverage but still have an RMS of on order of 1% as shown in the lower plot of each figure.

The other glaringly large offsets occur in plane 110 and plane 189 (using offline software numbering conventions). These regions are detailed in Fig. 5.

In the case of plane 110 the second module fits to be ∼3.5 cm lower than the initial geometry puts it and for plane 189 the first module fits to be ∼2.5 cm too low.

Additionally, as has been found in previous per-plane alignments, plane 52 is found to fit to about 1 cm low for all modules. This is shown in Fig. 6.

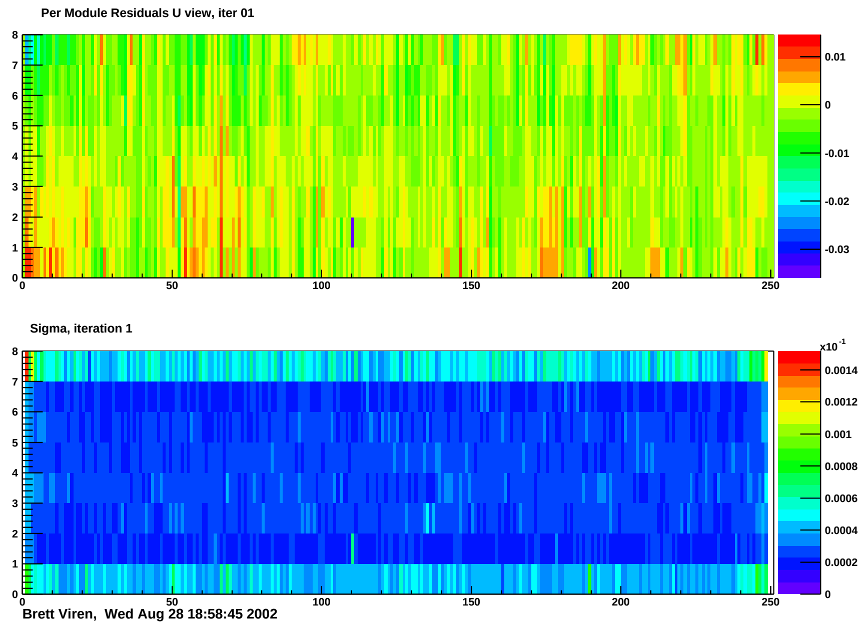

Figure 3: Per module mean and RMS fit residuals after second iteration. See text.

5 Future work

The above is an initial result and there is room for further work.

1. The current algorithm considers each candidate digit in a track separately and uses the center of its strip as the location. A refinement is to collect all candidate digits from the track for a particular plane and use a charge weighted sum of strip centers to define the transverse location of the track crossing.
2. Investigate the effect of further iterating the algorithm beyond three steps.
3. Since cosmic rays muons tend to differ from beam induced muons in direction and energy systematic effects need to be looked at. For example, finding out if the alignment changes with differing track length, angle, whether or not the track stops in the detector, etc.
4. Extend this to curved (B-field on) muon tracks and to per-strip alignment.
5. Determine at just what level we need to know the alignment of the detector. That is, investigate reconstruction as a function of different levels of alignment.
6. Compare alignmnet results to those of the survey data.

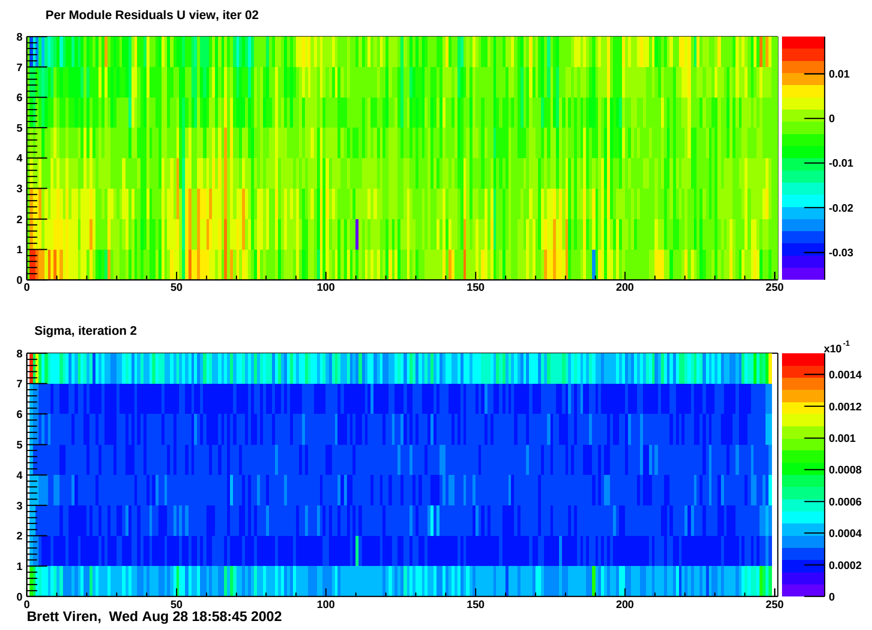

Figure 4: Per module mean and RMS fit residuals after third iteration. See text.

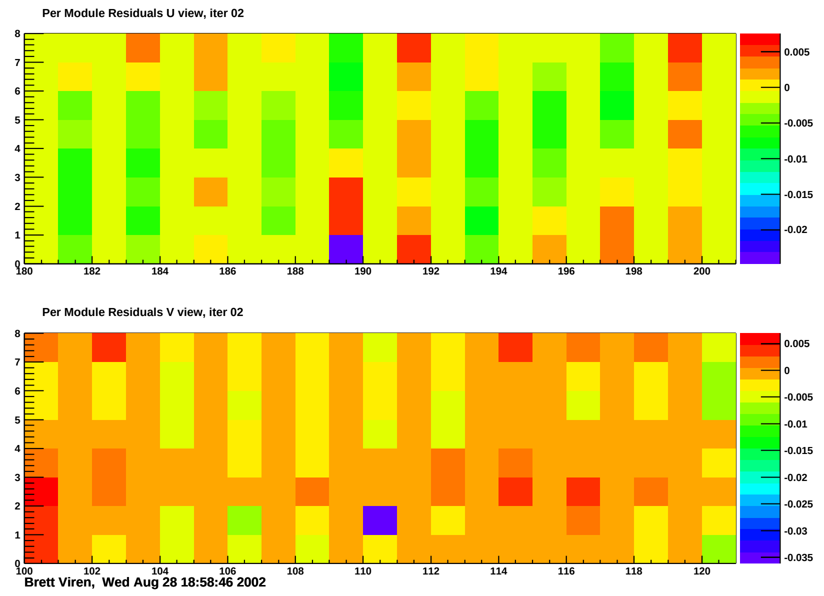

Figure 5: Zoom in on region of U and V views which have modules with large offsets.

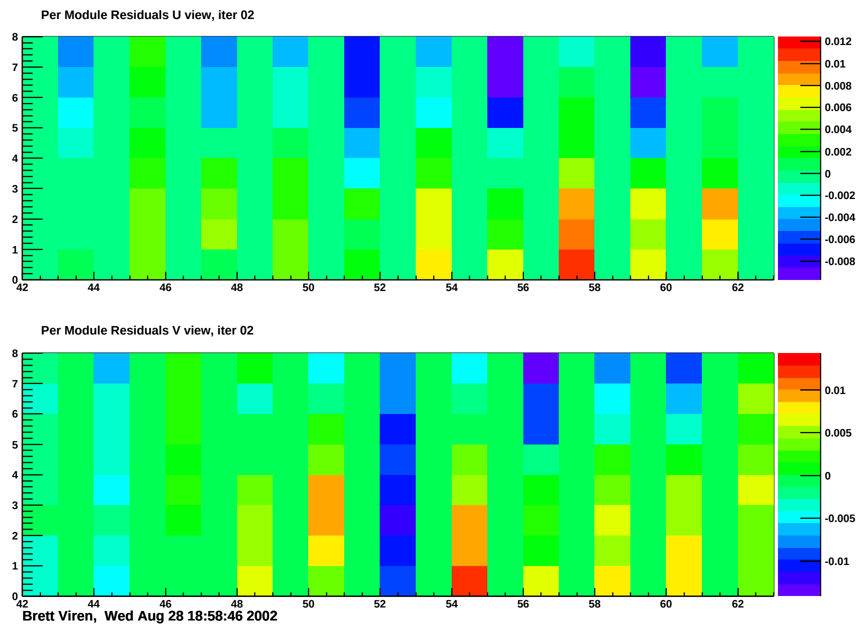

Figure 6: Closeup showing all modules in plane 52 fitting to be about 1 cm low.

7. Run the alignment after the survey data has been applied to the offline software geometry.

8. Fold these alignment offsets back into the offline software geometry.